**NỘI DUNG THI**

**I. CÂU HỎI LÝ THUYẾT**

# Biztalk Server là gì ? Ưu và nhược điểm của Biztalk Server ?

**A, Khái niệm**

**Biztalk Server là một hệ thống phần mêm tổ chức trung gian, cho phép các công ty tự động hóa các quy trình kinh doanh** thông qua việc sử dụng các bộ điều phối (adapters) phù hợp để giao tiếp với những hệ thống phần mềm khác nhau trong doanh nghiệp.

🡺 Chức năng:

+ Tích hợp ứng dụng doanh nghiệp ( EAI): kết nối các chức năng kinh doanh của một doanh nghiệp, thường bị phân tán trên các hệ (*platform*) khác nhau, nhằm hợp nhất các chu trình doanh nghiệp.

+ Tự động hóa các quy đình nghiệp vụ

+ Giao tiếp giữa các nghiệp vụ B2B.

+ Trung gian tin nhắn ( message broker)

+ Giảm sát hoạt động kinh doanh ( activiti morning)

**B, Ưu và nhược điểm.**

\*Ưu

- Biztalk Server là 1 công cụ và cơ sở hạ tấng cho tích hợp hệ thống.

- Đơn giản hóa và tự động hóa khả năng tương tác để giảm chi phí và lỗi

- Tự đống hóa các tương tác kinh doanh của bạn với các đối tác

- Đạt được những hiểu biết quan trọng về quy trình và hiệu suất kinh doanh.

- Các giải pháp có thể được phát triển rất nhanh với effortbor ra ít hơn.

- Cập nhật và bảo trì có thể diễn ra dễ dàng.

- Kịch bản tiêu cực có thể đc xử lý dễ dàng.

\*Nhược

-

-

-

# Trình bày về qui trình chuyển đổi và gửi thông điệp trong Biztalk Server

1. Thông điệp vào Hệ thống Biztalk Server thông qua một công nhận (Recevice Port). Mỗi công nhận có một hoặc nhiều địa điểm nhận. Mỗi địa điểm nhận được cấu hình bởi một bộ chuyển đổi (Adapter) dùng để xác định phương thức giao tiếp dùng để kết nối và nhận dữ liệu từ một hệ thống hoặc một ứng dụng bên ngoài, chẳng hạn như: 1 thư mục các tệp tin, 1 trang web HTTP, 1 cơ sở dữ | liệu SQL, hoặc ứng dụng của 1 bên thứ ba.

2. Thông điệp nhận được được xử lý bởi một ống nhận (Recevice Pipeline). Một Recevice Pipeline có chứa nhiều thành phần với các tính năng như: giải mã thông điệp, chia nhóm các thông điệp, chuyền thông điệp từ định dạng ban đầu sang định dạng XML, hoặc có thể chứng thực chữ ký số trên thông điệp.

3. Các cổng nhận có thể được cấu hình theo một hoặc nhiều Biểu đồ dữ | liệu (Map), Biều đồ dữ liệu cho phép chuyền thông điệp dữ liệu với cấu trúc này sang một cấu trúc khác. Biểu đồ dữ liệu được sử dụng để chuyển đổi các thông điệp từ nhiều định dạng khác nhau sang định dạng nội bộ hoặc định dạng được sử dụng thống nhất trong Hệ thống BizTalk.

4. Sau đó, thông điệp này sẽ được chuyển đến cơ sở dữ liệu Microsoft SQL Server, còn được gọi là Hộp chứa thông điệp (Message Box). Khi thông điệp | đến Hộp chứa thông điệp, Hệ thống thông qua những thành phần dữ liệu liên quan đến thông điệp để xác định những cổng hoặc những Orchestration liên quan mà thông điệp được chuyển đến.

5. Một Orchestration mô tả logic vận hành của một quy trình công việc. Mỗi một quy trình có thể được mô tả bởi một hoặc nhiều Orchestration. Mỗi một Orchestration bao gồm nhiều hình liên kết với nhau, mỗi hình có một ý nghĩa riêng và dùng để mô tả điều kiện, vòng lặp, các thói quen, hoạt động trong một quy trình.

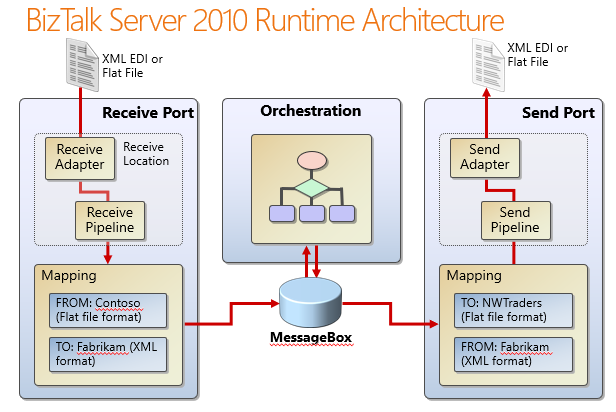
6. Một thông điệp có thể được xử lý bởi một Orchestration hoặc không, sau đó được chuyển đến Công gửi (Send Port). Cổng gửi có thể sử dụng Bản đồ dữ liệu để chuyển định dạng thông điệp và tiến hành xử lý thông điệp tại ống gửi (Send Pipeline).

7. Ống gửi (Send Pipeline) có thể chuyển thông điệp từ định dạng XML sử dụng trong Biztalk Server sang định dạng nơi cần chuyển đến, đồng thời tiến hành mã hóa thông điệp nhằm đảm bản an toàn thông tin trong quá trình truyền thông điệp.

8. Sau đó, Send Pipeline sử dụng Biểu đồ để kết nối và truyền dữ liệu sang hệ thống hoặc ứng dụng bên ngoài.

Phần Thông điệp (Messaging) của Biztalk Server có thể giao tiếp được với nhiều hệ thống và ứng dụng khác nhau. Nó hỗ trợ việc chuyển đổi dữ liệu từ định dạng này sang định dạng khác dựa trên các chuẩn.

# Trình bày về BizTalk Server Runtime Architecture



\* BizTalk Schemas: Mỗi loại tài liệu duy nhất yêu cầu một lược đồ riêng xác định các bản ghi và các trường có trong tài liệu đó. Lược đồ XML định nghĩa:

- Các yếu tố, thuộc tính và kiểu dữ liệu xuất hiện trong tài liệu

- Thứ tự của các thẻ trong tài liệu

- Các trường bắt buộc hoặc có thể xảy ra nhiều lần trong một tài liệu

\* BizTalk Map: là một tệp xác định các trường sự tương ứng giữa các trường trong hai lược đồ khác nhau.

- Chuyển đổi dữ liệu: Thực hiện các hoạt động tính toán + điều hpoois dữ liệu, copy dữ liệu từ tin nhắn này sang tin nhắn khác.

- Dịch dữ liệu: ( Chuyển định dạng của dữ liệu giữa các tin nhắn) ví dụ từ Flat file🡺 XML file.

\* Pipeline: là các thành phần phần mềm có thể xử lý tin nhắn, khi tin nhắn được nhận hoặc ngay trước khi chúng được gửi qua cổng gửi

-

- Chuẩn hóa dữ liệu từ các định dạng khác nhau sang XML (receive)

- Dịch dữ liệu từ XML sang các định dạng khác nhau (send)

- Lắp ráp và tháo rời các tài liệu

- Giải mã và mã hóa tài liệu

- Chỉ định và xác minh chữ ký số

\* Orchestration

- Là một bộ điều phối cung cấp các dịch vụ để thiết kế, thực hiện và quản lý cá quy trình kinh doanh. Một Orchestration mô tả logic vận hành của một quy trình công việc. Mỗi một quy trình có thể được mô tả bởi một hoặc nhiều Orchestration. Mỗi một Orchestration bao gồm nhiều hình liên kết với nhau, mỗi hình có một ý nghĩa riêng và dùng để mô tả điều kiện, vòng lặp, các thói quen, hoạt động trong một quy trình

\* Adapter: là thành phần phần mềm đầu tiên và cuối cùng được BizTalk Server sử dụng để trao đổi tin nhắn với các ứng dụng và hệ thống khác nhau bằng cơ chế gửi cụ thể như HTTP, FTP hoặc POP3.

\* Message box: là bộ phận lưu trữ các thông điệp và thuộc tính tin nhắn và nó duy trì việc đăng ký tin nhắn của người nhận. Công cụ nhắn tin chuyển các tin nhắn từ cơ sở dữ liệu đến các thuê bao. Cơ sở dữ liệu MessageBox cũng lưu trữ các hàng đợi và bảng trạng thái cho mỗi máy chủ BizTalk.

# Dịch vụ web là gì ? Ưu nhược điểm của dịch vụ web? Các ứng dụng của dịch vụ web

**Web service là**tập hợp những thành phần ứng dụng, để chuyển đổi bất kỳ 1 ứng dụng thông thường sang 1 ứng dụng web. Bên cạnh đó, nó cũng xuất bản những chức năng của mình để cho mọi người dùng internet đều sử dụng được thông qua nền tảng của web….**Web service**có thể được tạo ra bởi bất kể 1 ngôn ngữ lập trình nào đó.

Web service giúp cho client và server tương tác được với nhau ngay cả khi môi trường của chúng khác nhau. Chẳng hạn như: Đặt 1 website server cho 1 ứng dụng trên máy chủ hệ điều hành linux, trong khi đó người dùng lại dùng hệ điều hành windows. Tuy nhiên ứng dụng vẫn có thể chạy được và xử lý bình thường mà không cần thêm 1 yêu cầu đặc biệt nào tương thích giữa 2 hệ điều hành này.

Hầu như kỹ thuật của website service đều được xây dựng từ mã nguồn mở như là XML chẳng hạn. Website  service có thể bao gồm  nhiều mô-đun và có thể công bố trên hệ thống internet. Ứng dụng khi được phát triển sẽ hoạt động theo Client server.

Một dịch vụ web bao gồm có nhiều mô-đun và có thể công bố lên mạng Internet.

**ƯU VÀ NHƯỢC ĐIỂM CỦA WEB SERVICE LÀ GÌ?**

**ƯU ĐIỂM**

* web service cung cấp nền tảng rộng lớn chạy được trên các hệ điều hành khác nhau.
* Nâng khả năng tái sử dụng.
* Thúc đẩy đầu tư toàn bộ các hệ thống phần mềm đã tồn.
* Thúc đẩy mạnh mẽ vào hệ thống tích hợp. Và giảm được sự phức tạp của hệ thống, giảm giá thành hoạt động.
* Phát triển hệ thống nhanh và tương tác hiệu quả với hệ thống cảu các doanh nghiệp khác.
* Tạo ra được các mối quan hệ tương tác với nhau. Và sự qua lại giữa các thành phần trong hệ thống. Dễ dàng cho việc phát triển các ứng dụng phân tán.
* Dịch vụ Web cung cấp khả năng hoạt động rộng lớn với các ứng dụng phần mềm khác nhau chạy trên những nền tảng khác nhau.
* Sử dụng các giao thức và chuẩn mở. Giao thức và định dạng dữ liệu dựa trên văn bản (text), giúp các lập trình viên dễ dàng hiểu được.

#### NHƯỢC ĐIỂM:

* Có thể xảy ra thiệt hại không hề nhỏ vào khoảng thời gian chết của web service như: giao diện không thay đổi, có thể lỗi nếu một máy khách không được nâng cấp, thiếu các giao thức cho việc vận hành.
* Có quá nhiều tiêu chuẩn cho 1 web service khiến người dùng khó nắm bắt.
* Phải quan tâm nhiều hơn đến vấn đề an toàn và bảo mật.

**Các ứng dụng của web services**

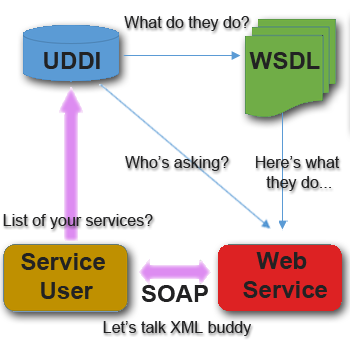
Tích hợp các hệ thống khác nhau trên các nền tảng khác nhau

Cơ bản tất cả những thứ có trong ưu điểm

# Trình bày các thành phần chính khi sử dụng dịch vụ web và giải thích các thành phần.

Nền tảng cơ bản của web service (WS) là XML + HTTP. Nó bao gồm các thành phần chính sau:

* **UDDI** – Universal Description, Discovery, and Integration (Mô tả, Khám phá và Tích hợp Toàn cầu): UDDI là một tiêu chuẩn dựa trên XML để mô tả, xuất bản và tìm kiếm các dịch vụ web.
* **WSDL** – Web Service Description Language (Ngôn ngữ mô tả web service): WSDL là một ngôn ngữ dựa trên XML để mô tả các dịch vụ web và cách truy cập chúng. WSDL mô tả một dịch vụ web, cùng với định dạng thông báo và các chi tiết giao thức cho dịch vụ web.
* **SOAP** – Simple Object Access Protocol (Giao thức truy cập đối tượng đơn giản): SOAP là một giao thức dựa trên XML đơn giản cho phép các ứng dụng trao đổi thông tin qua HTTP.

[](https://gpcoder.com/wp-content/uploads/2019/05/web-service-diagram.png)

### XML – eXtensible Markup Language

Là một chuẩn mở do W3C đưa ra cho cách thức mô tả dữ liệu, nó được sử dụng để định nghĩa các thành phần dữ liệu trên trang web và cho những tài liệu B2B. Về hình thức, XML hoàn toàn có cấu trúc thẻ giống như ngôn ngữ HTML nhưng **HTML định nghĩa thành phần được hiển thị như thế nào thì XML lại định nghĩa những thành phần đó chứa cái gì**. Với XML, các thẻ có thể được lập trình viên tự tạo ra trên mỗi trang web và được chọn là định dạng thông điệp chuẩn bởi tính phổ biến và hiệu quả mã nguồn mở.

Do dịch vụ Web là sự kết hợp của nhiều thành phần khác nhau nên nó sử dụng các tính năng và đặc trưng của các thành phần đó để giao tiếp. XML là công cụ chính để giải quyết vấn đề này và là kiến trúc nền tảng cho việc xây dựng một dịch vụ Web, tất cả dữ liệu sẽ được chuyển sang định dạng thẻ XML. Khi đó, các thông tin mã hóa sẽ hoàn toàn phù hợp với các thông tin theo chuẩn của SOAP hoặc XML-RPC và có thể tương tác với nhau trong một thể thống nhất.

### WSDL – Web Service Description Language

WSDL định nghĩa cách mô tả dịch vụ Web theo cú pháp tổng quát của XML, bao gồm các thông tin:

* Tên dịch vụ.
* Giao thức và kiểu mã hóa sẽ được sử dụng khi gọi các hàm của dịch vụ Web.
* Loại thông tin: thao tác, tham số, những kiểu dữ liệu (có thể là giao diện của dịch vụ Web cộng với tên cho giao diện này).

Một WSDL hợp lệ gồm hai phần: phần giao diện mô tả giao diện và phương thức kết nối và phần thi hành mô tả thông tin truy xuất CSDL. Cả hai phần này sẽ được lưu trong 2 tập tin XML tương ứng là tập tin giao diện dịch vụ và tập tin thi hành dịch vụ. Giao diện của một dịch vụ Web được miêu tả trong phần này đưa ra cách thức làm thế nào để giao tiếp qua dịch vụ Web. Tên, giao thức liên kết và định dạng thông điệp yêu cầu để tương tác với dịch vụ Web được đưa vào thư mục của WSDL.

WSDL thường được sử dụng kết hợp với XML schema và SOAP để cung cấp dịch vụ Web qua Internet. Một client khi kết nối tới dịch vụ Web có thể đọc WSDL để xác định những chức năng sẵn có trên server. Sau đó, client có thể sử dụng SOAP để lấy ra chức năng chính xác có trong WSDL.

### Universal Description, Discovery, and Integration (UDDI)

Để có thể sử dụng các dịch vụ, trước tiên client phải tìm dịch vụ, ghi nhận thông tin về cách sử dụng và biết được đối tượng nào cung cấp dịch vụ. UDDI định nghĩa một số thành phần cho biết các thông tin này, cho phép các client truy tìm và nhận những thông tin được yêu cầu khi sử dụng dịch vụ Web.

Cấu trúc UDDI :

* Trang trắng – White pages: chứa thông tin liên hệ và các định dạng chính yếu của dịch vụ Web, chẳng hạn tên giao dịch, địa chỉ, thông tin nhận dạng… Những thông tin này cho phép các đối tượng khác xác định được dịch vụ.
* Trang vàng – Yellow pages: chứa thông tin mô tả dịch vụ Web theo những loại khác nhau. Những thông tin này cho phép các đối tượng thấy được dịch vụ Web theo từng loại với nó.
* Trang xanh – Green pages: chứa thông tin kỹ thuật mô tả các hành vi và các chức năng của dịch vụ Web.
* Loại dịch vụ – tModel:  chứa các thông tin về loại dịch vụ được sử dụng.

Những thông tin về dịch vụ Web được sử dụng và công bố lên mạng sử dụng giao thức này. Nó sẽ kích hoạt các ứng dụng để tìm kiếm thông tin của dịch vụ Web khác nhằm xác định xem dịch vụ nào sẽ cần đến nó.

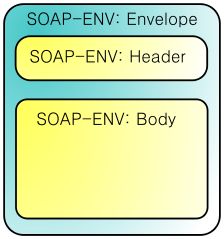
### SOAP – Simple Object Access Protocol

Chúng ta đã hiểu cơ bản dịch vụ Web như thế nào nhưng vẫn còn một vấn đề khá quan trọng. Đó là làm thế nào để truy xuất dịch vụ khi đã tìm thấy? Câu trả lời là các dịch vụ Web có thể truy xuất bằng một giao thức là Simple Object Access Protocol – SOAP. Nói cách khác chúng ta có thể truy xuất đến UDDI registry bằng các lệnh gọi hoàn toàn theo định dạng của SOAP.

SOAP là một giao thức giao tiếp có cấu trúc như XML. Nó được xem là cấu trúc xương sống của các ứng dụng phân tán được xây dựng từ nhiều ngôn ngữ và các hệ điều hành khác nhau. SOAP là giao thức thay đổi các thông điệp dựa trên XML qua mạng máy tính, thông thường sử dụng giao thức HTTP.

Một client sẽ gửi thông điệp yêu cầu tới server và ngay lập tức server sẽ gửi những thông điệp trả lời tới client. Cả SMTP và HTTP đều là những giao thức ở lớp ứng dụng của SOAP nhưng HTTP được sử dụng và chấp nhận rộng rãi hơn bởi ngày nay nó có thể làm việc rất tốt với cơ sở hạ tầng Internet.

**Cấu trúc một thông điệp theo dạng SOAP**

[](https://gpcoder.com/wp-content/uploads/2019/05/6-soap-structure.jpg)

Thông điệp theo định dạng SOAP là một văn bản XML bình thường bao gồm các phần tử sau:

* Phần tử gốc – **envelop**: phần tử bao trùm nội dung thông điệp, khai báo văn bản XML như là một thông điệp SOAP.
* Phần tử đầu trang – **header**: chứa các thông tin tiêu đề cho trang, phần tử này không bắt buộc khai báo trong văn bản. Header còn có thể mang những dữ liệu chứng thực, những chứ ký số, thông tin mã hóa hay cài đặt cho các giao dịch khác.
* Phần tử khai báo nội dung chính trong thông điệp – **body**, chứa các thông tin yêu cầu và thông tin được phản hồi.
* Phần tử đưa ra các thông tin về lỗi – **fault**, cung cấp thông tin lỗi xảy ra trong qúa trình xử lý thông điệp.

Một SOAP đơn giản trong body sẽ lưu các thông tin về tên thông điệp, tham chiếu tới một thể hiện của dịch vụ, một hoặc nhiều tham số. Có 3 kiểu thông báo sẽ được đưa ra khi truyền thông tin: request message(tham số gọi thực thi một thông điệp), respond message (các tham số trả về, được sử dụng khi yêu cầu được đáp ứng) và cuối cùng là fault message (thông báo tình trạng lỗi).

Ví dụ:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <?xml version="1.0"?>  <soap:Envelope xmlns:soap="<http://www.w3.org/2001/12/soap-envelope>" soap:encodingStyle="<http://www.w3.org/2001/12/soap-encoding>">    <soap:Header>  ...  </soap:Header>    <soap:Body>  ...    <soap:Fault>    ...    </soap:Fault>  </soap:Body>    </soap:Envelope> |

**Kiểu truyền thông**

Có 2 kiểu truyền thông

* Remote procedure call (RPC): cho phép gọi hàm hoặc thủ tục qua mạng. Kiểu này được khai thác bởi nhiều dịch vụ Web.
* Document: được biết đến như kiểu hướng thông điệp, nó cung cấp giao tiếp ở mức trừu tượng thấp, khó hiểu và yêu cầu lập trình viên mất công sức hơn.

Hai kiểu truyền thông này cung cấp các định dạng thông điệp, tham số, lời gọi đến các API khác nhau nên việc sử dụng chúng tùy thuộc vào thời gian và sự phù hợp với dịch vụ Web cần xây dựng.

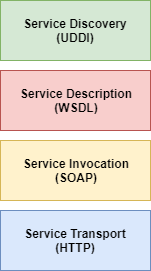
**Cấu trúc dữ liệu**

Cung cấp những định dạng và khái niệm cơ bản giống như trong các ngôn ngữ lập trình khác như kiểu dữ liệu (int, string, date…) hay những kiều phức tạp hơn như struct, array, vector… Định nghĩa cấu trúc dữ liệu SOAP được đặt trong namespace SOAP-ENC.

**Mã hóa**

Giả sử service requester và service provider được phát triển trong Java, khi đó mã hóa SOAP là làm thế nào chuyển đổi từ cấu trúc dữ liệu Java sang SOAP XML và ngược lại, bởi vì định dạng cho Web Service chính là XML. Bất kỳ một môi trường thực thi SOAP nào cũng phải có một bảng chứa thông tin ánh xạ nhằm chuyển đổi từ ngôn ngữ Java sang XML và từ XML sang Java – bảng đó được gọi là SOAPMappingRegistry. Nếu một kiểu dữ liệu được sử dụng dưới một dạng mã hóa thì sẽ có một ánh xạ tồn tại trong bộ đăng ký của môi trường thực thi SOAP đó.

## Web Services Architecture

[](https://gpcoder.com/wp-content/uploads/2019/05/Web-Services-Architecture-1.png)

* **Service Discovery** : Phần kiến ​​trúc này chịu trách nhiệm tập trung các dịch vụ vào một nơi đăng ký chung và cung cấp chức năng publish/ search dễ dàng. Điều này được xử lý bởi **UDDI**.
* **Service Description** : Một trong những tính năng thú vị nhất của Dịch vụ web là chúng tự mô tả. Điều này có nghĩa là, một khi Dịch vụ web được định vị, nó sẽ cho chúng ta biết những hoạt động mà nó hỗ trợ và cách gọi nó. Điều này được xử lý bởi **WSDL**.
* **Service Invocation** : Gọi một dịch vụ web liên quan đến việc truyền tin nhắn giữa Client và Server. **SOAP** chỉ định cách chúng ta nên định dạng các thông báo yêu cầu (request) đến Server và cách Server định dạng các thông điệp phản hồi (response) của nó.
* **Service Transport** : Cuối cùng, tất cả các thông báo này phải được truyền đi bằng cách nào đó giữa Client và Server. Giao thức được lựa chọn cho phần kiến ​​trúc này là **HTTP** – giao thức được sử dụng để truy cập các trang web thông thường trên Internet. Chúng ta cũng có thể sử dụng các giao thức khác, nhưng HTTP hiện là giao thức được sử dụng nhiều nhất.

# EAI là gì ? Các dạng EAI ? Các mức độ của EAI

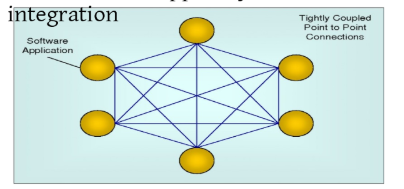
*- Enterprise application integration – EAI*:  là một khung tích hợp bao gồm một tập hợp các công nghệ, dịch vụ hình thành 1 trung gian để cho phép tích hợp các hệ thống và các ứng dụng độc lập ( bao gồm các hệ thống bên ngoài  liên quan đên chu trình của doanh nghiệp) trên toàn doanh nghiệp

- Tích hợp ứng dụng doanh nghiệp là quá trình liên kết các ứng dụng với nhau trong 1 tổ chức duy nhất để đơn giản hóa và tự động hóa quy trình kinh doanh đến mức lớn nhất có thể, trong khi đồng thời tránh việc phải thực hiện thay đổi sâu rông đến các ứng dựng hiện tại hoặc cấu trức dữ liệu

- EAI không hạn chế chia sẻ dữ liệu và quy trình kinh doanh trong bất kỳ ứng dựng kết nối hoặc các nguồn dữ liệu doanh nghiệp

- Các dạng EAI:

+ Point to point: kết nối điểm với điểm. Trình kết nối cho phép ứng dụng A giao tiếp trực tiếp với ứng dụng B.



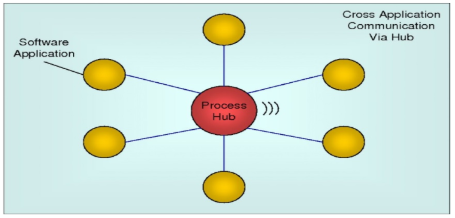
Ưu điểm:- Thích hợp cho những hệ thống nhỏ.

* Nhanh và hiệu quả.
* Các ứng dụng liên kết chặt chẽ với nhau

Nhược điểm:

* Độ phức tạp tăng khi ứng dụng tăng.
* Mức độ bảo trì cao
* Thiếu linh hoạt do sự liên kết chặt chẽ, khó trong việc quản lý

+ Hub and Spoke: Hoạt động theo nguyên tắc tất cả thông tin đến từ các ứng dụng phải được xử lý trong một máy hoặc một máy chủ gọi là trung tâm



Ưu điểm: Ít phức tạp hơn p-p đối với hệ thống lớn

Nhược điểm:

* Có quá nhiều xử lý diễn ra ở trung tâm
* Khi số lượng và độ phức tạp của thuật toán tăng lên hiệu xuất có thể bị ảnh hưởng và các trung tâm khó quản lý, duy trì và mở rộng

+ Distributed Integration Architecture

+ Service Oriented Architecture

# Trình bày về kiến trúc Hub and Spoke trong EAI.

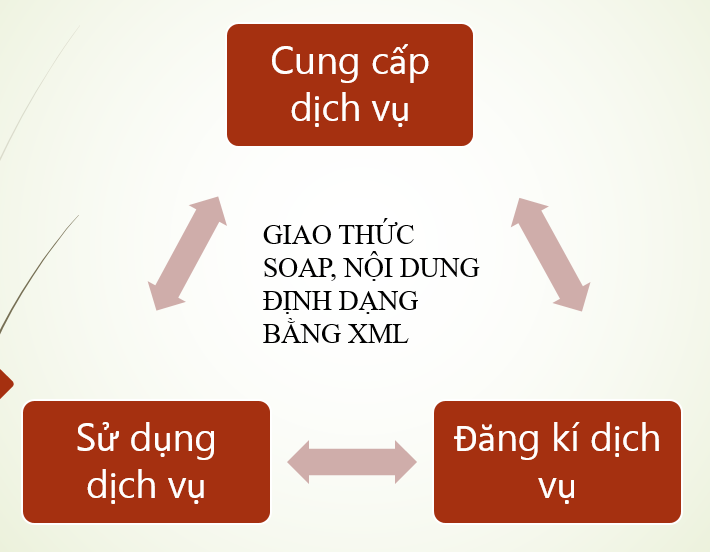
# Trình bày về kiến trúc tích hợp phân tán (Distributed Integration Architecture) trong EAI

# SOA là gì ? Các thành phần của SOA?

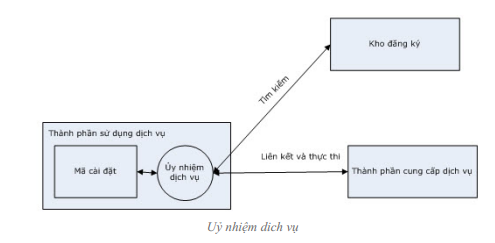
SOA là một hướng tiếp cận với việc thiết kế và tích hợp các phần mềm hay chức năng hệ thống theo dạng module, trong đó mỗi module đóng vai trò là một “**dịch vụ** có tính kết nối lỏng lẻo” và có khả năng truy cập thông qua môi trường mạng. SOA là hướng tiếp cận xây dựng phần mềm dựa trên sự kết hợp các dịch vụ

Một số ưu điểm của việc phát triển ứng dụng hướng dịch vụ (SOA)

* Thứ nhất, tái sử dụng phần mềm. Nếu một dịch vụ có quy mô và kích thước phù hợp sau đó nó có thể được tái sử dụng cho lần kế tiếp. Điều này đồng nghĩa sẽ làm giảm công sức phát triển và chi phí về mặt tài chính cho cả hai phía: nhà phát triển phần mềm và các khách hàng (doanh nghiệp).
* Thứ hai, linh hoạt khi mở rộng, kết nối và tích hợp. Giả sử rằng các dịch vụ sẽ không được tái sử dụng, thì ta có thể đưa ra nhiều giá trị nếu ta làm cho hệ thống CNTT chỉnh sửa dễ dàng hơn.
* Các thành phần của SOA
* Dịch vụ (Service):
  + Dịch vụ có tính chất rõ ràng, là một đơn vị chức năng nghiệp vụ có thể được triệu gọi
  + Có khả năng triệu gọi thông qua các giao thức truyền thông chung.
  + Có tính liên thông và vị trí trong suốt.
  + Dịch vụ được định nghĩa bằng các giao diện tường minh.
  + Các giao diện độc lập với cài đặt.
  + Cung cấp giao ước giữa các thành phần cung cấp và sử dụng dịch vụ.
  + Dịch vụ là các modul phức tạp, bao gồm nhiều thành phần. Mức độ đóng gói của dịch vụ càng cao thì dịch vụ càng có khả năng tái sử dụng và linh hoạt.
* Thành phần sử dụng dịch vụ (service/cusumer/client/request)
  + Là một ứng dụng, một dịch vụ, hoặc một loại module phần mềm khác có yêu cầu sử dụng dịch vụ.
  + Là thực thể khởi tạo việc định vị định vụ tại một kho đăng ký dịch vụ, liên kết tới dịch vụ qua một kênh truyền thông và thực thi các chức năng của dịch vụ.
  + Thành phần này thực thi nhiệm vụ bằng cách gửi tới dịch vụ một yêu cầu được định dạng theo đúng giao ước.
* Thành phần cung cấp dịch vụ (services provider)
  + là một thực thể có khả năng được địa chỉ hoá qua mạng, nó có thể chấp nhận và thực thi các yêu cầu từ những thành phần sử dụng dịch vụ.
  + Là một hệ thống máy tính lớn, một thành phần, hoặc loại hệ thống phần mềm khách có thể thực thi các yêu cầu dịch vụ.
  + Thực thể này xuất bản giao ước dịch vụ của nó trong một kho đăng ký dịch vụ để các thành phần sử dụng dịch vụ có thể truy cập.
* Thành phần đăng ký dịch vụ ( services registry)
  + Là một thư mục trên mạng có chứa các dịch vụ sẵn dùng.
  + Đây là một thực thể chấp nhận và lưu trữ các giao ước từ các thành phần cung cấp dịch vụ và cung cấp các giao ước đó cho các thành phần sử dụng dịch vụ.
  + Dịch vụ UDDI, đóng vai trò như người chỉ đường cho các thành phần sử dụng dịch vụ, tạo sự trong suốt về vị trí đối với thành phần cung cấp dịch vụ; để khi thành phần này thay đổi thì thành phần sử dụng dịch vụ không cần phải biên dịch hay cấu hình lại.



* Giao ước dịch vụ ( contract)
  + Một giao ước là một bản đặc tả cách thức để thành phần sử dụng dịch vụ có thể tương tác với thành phần cung cấp dịch vụ. Nó chỉ ra khuôn dạng của thông điệp yêu cầu và thông điệp đáp ứng từ các dịch vụ.
  + Giao ước dịch vụ có thể đòi hỏi một tập các điều kiện tiên quyết và điều kiện sau. Các điều kiện này xác định trạng thái cần thiết của dịch vụ để thực thi một chức năng cụ thể.
  + Bản giao ước này cũng có thể bao gồm các mức độ chất lượng của dịch vụ, các đặc tả cho các khoá chạnh phi chức năng của dịch vụ
* ủy nhiệm dịch vụ
  + Ủy nhiệm dịch vụ là một thực thể không bắt buộc, nó chỉ đơn giản hoá cho thành phần sử dụng dịch vụ và thành phần sử dụng dịch vụ hoàn toàn có thể viết phần mềm để truy cập tới dịch vụ.
  + Một thành phần cung cấp dịch vụ sẽ cung cấp nhiều uỷ nhiệm cho các môi trường khác nhau, mỗi uỷ nhiệm dịch vụ được viết bằng ngôn ngôn ngữ của các thành phần sử dụng dịch vụ
  + Ví dụ: Một thành phần cung cấp dịch vụ có thể cung cấp các uỷ nhiệm dịch vụ cho Java, Visual Basic, Delphi nếu đó là các nền tảng của các thành phần sử dụng dịch vụ. Mặc dù ủy nhiệm dịch vụ là không bắt buộc nhưng có thể cải thiện một cách đáng kể hiệu năng và tính tiện dụng cho các thành phần sử dụng dịch vụ



* ràng buộc sử dụng dịch vụ ( services lease)

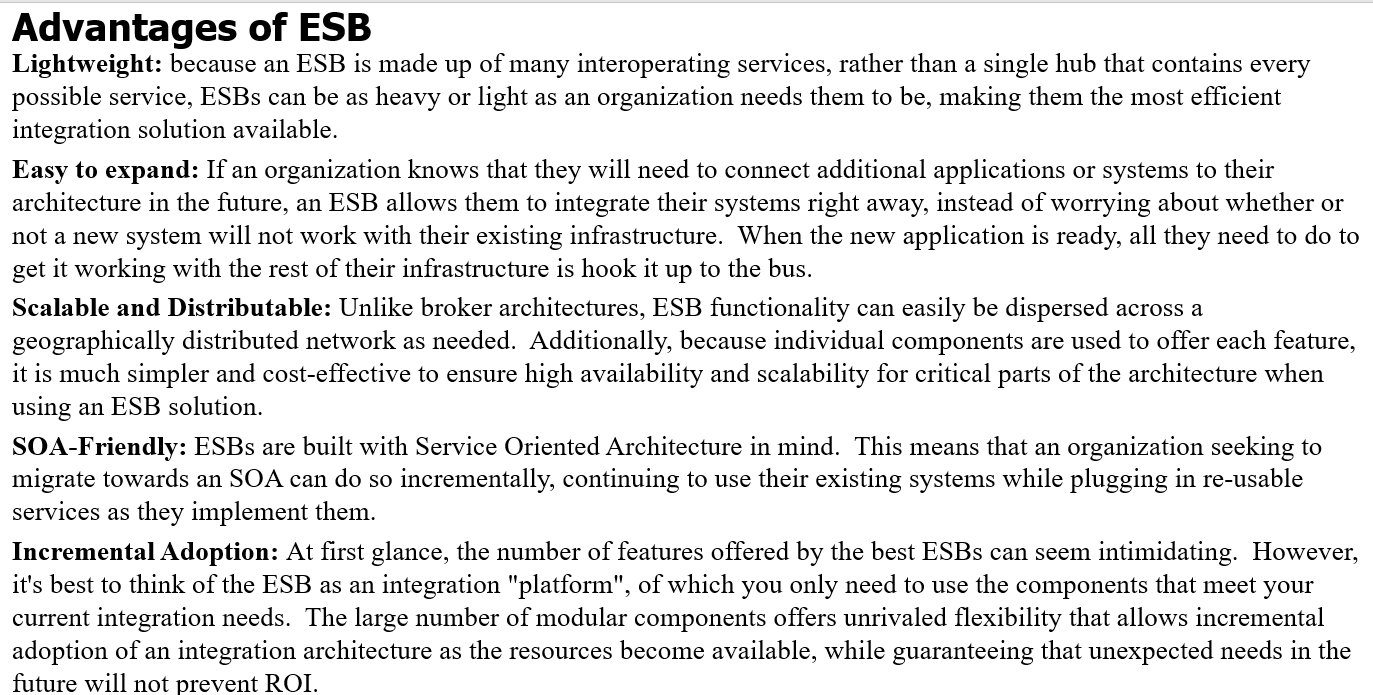
Ràng buộc sử dụng dịch vụ mà các thành phần đăng ký dịch vụ gán cho thành phần sử dụng dịch vụ rất cần thiết để dịch vụ bảo trì được thông tin trạng thái liên kết giữa thành phần sử dụng và thành phần cung cấp. Nó tạo ra sự gắn kết không chặt chẽ giữa các thành phần này bằng cách giới hạn khoảng thời gian mà chúng được liên kết với nhau. Không ràng buộc, một thành phần sử dụng dịch vụ có thể liên kết với một dịch vụ mãi mãi và không bao giờ liên kết lại với các giao ước của nó

# ESB là gì ? Ưu và nhược điểm của ESB ?

ESB là một hệ thống giải quyết các vấn đề liên quan đến việc tích hợp mà không cần phải xóa bỏ những giải pháp có sẵn. Mục đích của ESB là làm cho việc tích hợp các ứng dụng và quy trình trở nên thuận tiện hơn bằng cách cung cấp một quy trình phân tán, điều hướng thông minh, bảo mật và có thể tự động chuyển đổi dữ liệu. Trong hệ thống ESB những dịch vụ trên là những dịch vụ nền tảng do đó các ứng dụng không cần phải thi hành riêng biệt những yêu cầu trên theo một cách thức riêng biệt của chúng.

Ưu điểm:

Ứng dụng cuối gọn gàng do không có nhiều mã liên quan đến tích hợp. Phù hợp với mô hình component, các ứng dụng tích hợp lỏng lẻo qua ESB. Cho phép tích hợp với các ứng dụng với nền tảng lạc hậu. Dễ dàng đáp ứng cho việc trao đổi dữ liệu giữa các ứng dụng và đảm bảo tính nhất quán. Đễ dàng kiểm soát các hoạt động, quản lí  tập trung.



Nhược điểm:

Chi trao đổi đối với tệp cấp thấp, dữ liệu thô. Dễ bị thắt cổ chai do truyền nhận hoặc do số lượng tích hợp tăng lên. Cần phải thiết kế và resize cẩn thận.

Có ít chuyên gia giỏi.

All ứng dụng chung 1 hệ thống, khi có 1 thay đổi đối với esb đều gây ra nguy cơ đối với tất cả các ứng dụng tích hợp vào hệ thống.

Cần phải triển khai đúng cách, nhưng nếu không được thiết kế đúng hoặc có sai sót, độ phức tạp của hệ thống tăng rất nhanh. Đừng dùng ESB khi bạn bắt buộc(Mule CTO cảnh báo).

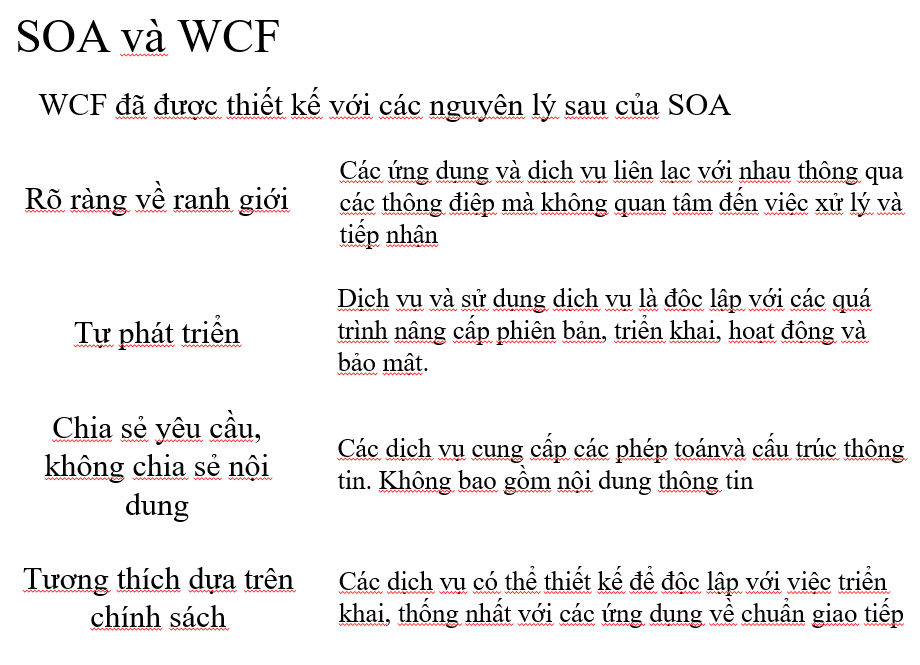
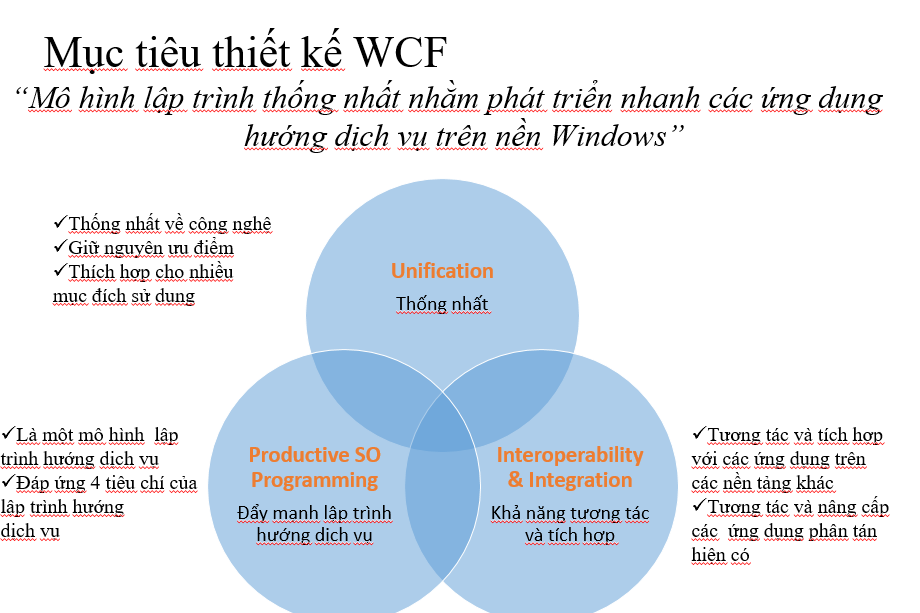
Không phù hợp với các hệ thống có yêu cầu thay đổi nhanh, đáp ứng cho hoạt động kinh doanh. Trong khi đó xu hướng hiện tại việc tổ chức hệ thống xung quanh nhu cầu của kinh doanh hơn là khả năng CNTT. ESB bị mircroservice thay thế.

# Trình bày các tính năng chính của ESB ?

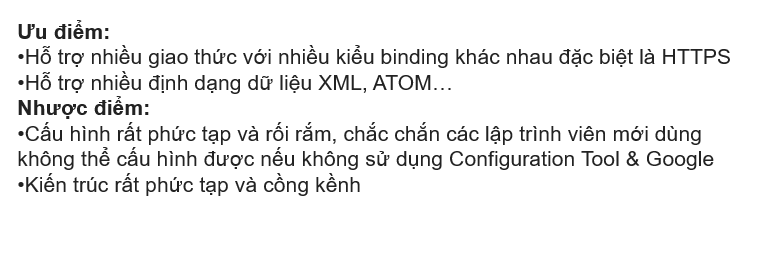
* Location transparency
* Transformation – usable formats for all consumers
* Protocol conversion – Accept all protocols for consumption
* Routing – determine appropriate end consumer based on preconfigured rules or dynamic created requests.
* Monitoring/Administration
* Security – ESB security involves tow main components
* Ensure secure handing of messages
* Ensure secure transport of messages
* Connecting Anything to Anything
  + Transports: HTTP, HTTPS, POP, IMAP, SMTP, JMS, AMQP, FIX, TCP, UDP, FTPS, SFTP, SMS
  + Formats & protocols: JSON, XML, SOAP 1.1, SOAP 1.2, WS-\*, HTML, EDI, HL7
* Routing, Mediation & Transformation
  + Routing: Header based, content based, rule-based and priority-based routing
  + Mediation: EIPs (including scatter/gather, message filters, recipient list, dead-letter channels, guaranteed delivery and message enrichment), database integration, event publishing, logging & auditing, validation
  + Transformation: XSLT 1.0/2.0, XPath, XQuery, Smooks
* Message, Service, API & Security Gateway
  + Expose existing applications & services over different protocols & message formats
  + Virtualize services for loose coupling & SOA governance
  + Load balancing for scalability and fail-over for high availability of business endpoints
  + Create service facades for legacy / non-standard services
  + Enforce and manage security centrally, including authentication, authorization & entitlement
  + Policy enforcement and governance via Policy Governance Registry
  + Expose services & applications via RESTful APIs with key management
  + Logging, audit and SLA monitoring, KPI monitoring
  + WS-Security, LDAP, Kerberos, OpenID, SAML, XACML
  + SSL tunneling and SSL profiles support for inbound and outbound scenarios
  + CRL/OCSP Certificate revocation verification

# WCF là gì ? Ưu và nhược điểm của WCF ? Trình bày về các thành phần Contracts trong WCF.

* WCF – (Windows Communication Foundation) là công nghệ mới nhất được thiết kế theo kiến trúc hướng dịch vụ (Service Oriented Architecture – SOA). Mục đích của WCF là tạo một công nghệ thống nhất cho các công nghệ giao tiếp trước đây như: Web Services, .NET Remoting,  Microsoft Message Queuing và COM+.
* Ba thành phần chính của một WCF service là:
* – **Service class**: định nghĩa các contact
* – **Hosting environment**:  WCF service có thể được host lên nhiều “môi trường” khác nhau, bao gồm IIS, Windows service, Selt-hosting.
* – **End point**: Các “cổng” kết nối giữa client và service.

* Ưu điểm:
  + của nó làm bằng rất nhiều thành phần khác nhau, vì vậy bạn có thể tạo ra thành phần mới cho an ninh, xác thực
  + của nó nhanh hơn ASMX
  + Hỗ trợ Scaling/Load Balancing
  + Khả năng tương tác của nó, cho java, v.v.
  + Khả năng tương tác qua nhiều nền tảng
  + WCF, WWF, WPF là bộ khung chủ đạo cho việc phát triển các ứng dụng nền Windows
  + Cho phép phát triển các ứng dụng hướng dịch vụ dựa trên kết nối giữa các dịch vụ và các ứng dụng
  + Thay thế cho các kiến trúc phân tán trước đây như Web services, COM+, Microsoft Message Queue, or .NET Framework Remoting
  + Cung cấp mô hình thống nhất cho viết thiết kế và kiến trúc các giải pháp mà không bị giới hạn bởi một cơ cấu riêng nào đó



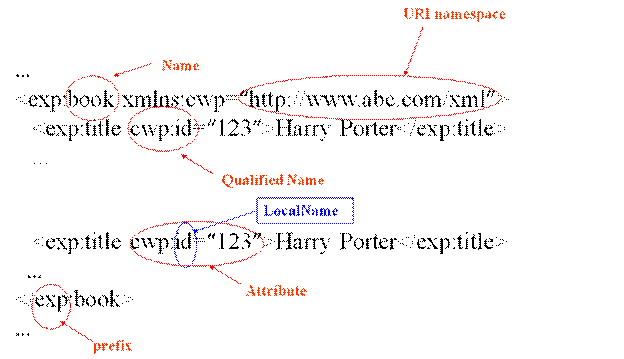
# XML là gì? Thế nào là một XML hợp khuôn dạng? Ưu nhược điểm của XML trong lưu trữ và truyền dữ liệu

* Xml là gì?

XML là viết tắt của từ **eXtensible Markup Language**, hay còn gọi là ngôn ngữ đánh dấu mở rộng do [W3C](http://www.w3.org/) đề nghị với mục đích tạo ra các ngôn ngữ đánh dấu khác. Đây là một tập hợp con đơn giản có thể mô tả nhiều loại dữ liệu khác nhau nên rất hữu ích trong việc chia sẻ dữ liệu giữa các hệ thống. Ví dụ khi bạn xây dựng một ứng dụng bằng C# và một ứng dụng bằng PHP thì hai ngôn ngữ này không thể hiểu nhau, vì vậy ta sẽ sử dụng XML để trao đổi dữ liệu.

Tất cả những đặc tả dữ liệu XML đều phải tuân theo quy luật và cú pháp của nó nên hầu như các file XML đều rất nghiêm khắc trong việc biên dịch. Tuy nhiên công nghệ này cần phải được xem xét bởi vì trong quá trình thao tác và truyền dữ liệu nó có tỉ lệ sai sót lên tới 5% - 7%. Con số này không cao nhưng cũng rất đáng để cân nhức khi sử dụng.

* XML hợp khuôn dạng hay là các quy tắc để viết XML đúng cú pháp
* Well-formed: các qui định tổng quát về cách thức viết XML đúng cú pháp
  + Tài liệu XML chỉ có duy nhất một tag làm root duy nhất và sẽ chứa tất cả các node con bên trong
    - Nội dung này thể hiện khái niệm tương tự như mô hình cây, chỉ có một nốt root duy nhất và là cha duy nhất trong cây, sau đó là các node con bên dưới
  + Các tag trong tài liệu xml phải tuân thủ qui luật là mở tag phải có đóng tag
  + <abc>…</abc>
  + Nếu là tag rỗng thì phải dùng dạng cú pháp <abc/>
  + Ký tự đầu tiên của tên luôn bắt đầu với một chữ hay dấu gạch dưới (\_)
  + Kế tiếp ký tự theo có thể là chữ, số, gạch dưới, gạch ngang, dấu chấm, và dấu chấm phẩy
  + Trong tên không cho phép bất kỳ ký tự khoảng trắng, enter, tab (whitespace) nào cả
  + Tên nên đặt có ý nghĩa và dễ dàng đọc hiểu
  + Các tên tag phải tuân theo qui luật đặt tên
  + Phải đặt tên tương tự qui luật của tên tag
  + Tên của các thuộc tính trong tag là duy nhất
  + Nếu tên thuộc tính có kết hợp với namespace (qualified name bao gồm namespace\_prefix:local\_name) thì cũng phải tuân theo qui luật nó là duy nhất
    - Tên namespace là giống nhau
    - Tên local name là khác nhau
  + Thuộc tính luôn xuất hiện một cặp bao gồm
  + tên\_thuộc\_tính = “giá trị của thuộc tính” hay
  + tên\_thuộc\_tính = ‘giá trị của thuộc tính’
  + Tất cả các tag trong tài liệu phải được lồng nhau theo đúng thứ tự, tag không được đóng mở tùy ý
  + Tất cả các thuộc tính trong tag
  + Tất cả khai báo namespace trong tài liệu XML phải có prefix khác nhau, duy nhất và tuân đúng luật về đặt tên tương tự như tag
  + XML là một ngôn ngữ có phân biệt hoa thường – case sensitive, do vậy, chúng ta cần lưu ý khi sử dụng ký tự trong quá trình đặc tên và truy xuất
  + Một ví dụ về qualified name của tag và attribute. Qua đó, chúng tôi mô tả trực tiếp từ khóa để dễ dàng đọc hiểu nội dung lý thuyết và các tài liệu của liên quan đến xml về qualified name



Ưu điểm:***.***

***1. XML tách biệt dữ liệu ra khỏi HTML :***

Khi cần hiển thị dữ liệu “động” trên văn bản HTML, nó sẽ rất mất nhiều thời gian để hiệu chỉnh lại trên HTMl mỗi khi dữ liệu cần cập nhật.

Với XML, dữ liệu có thể được trữ lại ở các file XML riêng biệt. Vì vậy, bạn có thể tập trung vào việc sử dụng HTML/CSS trong việc xây dựng layout mà vẫn đảm bảo rằng các thay đổi trong dữ liệu sẽ không gây ảnh hưởng gì.

***2. XML đơn giản hóa việc chia sẻ dữ liệu :***

Trong thực tế, hệ thống máy tính và cơ sở dữ liệu chứa các dữ liệu dưới những dạng khác nhau.

Dữ liệu XML được chứa dưới dạng text đơn giản, nó cho phép tạo ra các dữ liệu có thể được chia sẻ bằng các ứng dụng khác nhau.

***3. XML đơn giản hóa việc truyền dẫn dữ liệu :***

Một trong những công việc tốn thời gian nhất đó là trao đổi dữ liệu giữa các hệ thống khác nhau trên Internet.

Tuy nhiên, khi trao đổi dữ liệu dưới bằng XML sẽ đơn giản hóa mọi việc đi rất nhiều vì khi đó dữ liệu có thể đọc được trên nhiều hệ thống khác biệt.

***4. XML đơn giản hóa trong nâng cấp hệ thống :***

Khi nâng cấp hệ thống, một lượng lớn dữ liệu cần phải được chuyển hóa và có thể có một số bị mất do những khác biệt nhất điịnh.

XML chứa dữ liệu dưới dạng text, nó sẽ giúp cho mọi việc trở nên đơn giản hơn khi cần mở rộng hay nâng cấp hệ điều hành, ứng dụng, trình duyệt,… mà không làm mất đi dữ liệu.

***5. XML khiến cho dữ liệu dễ dàng được tiếp cận hơn :***

Nhiều ứng dụng khác nhau có thể tiếp cận với dữ liệu của bạn, không chỉ trên các trang HTML, mà còn các nguồn dữ liệu XMl.

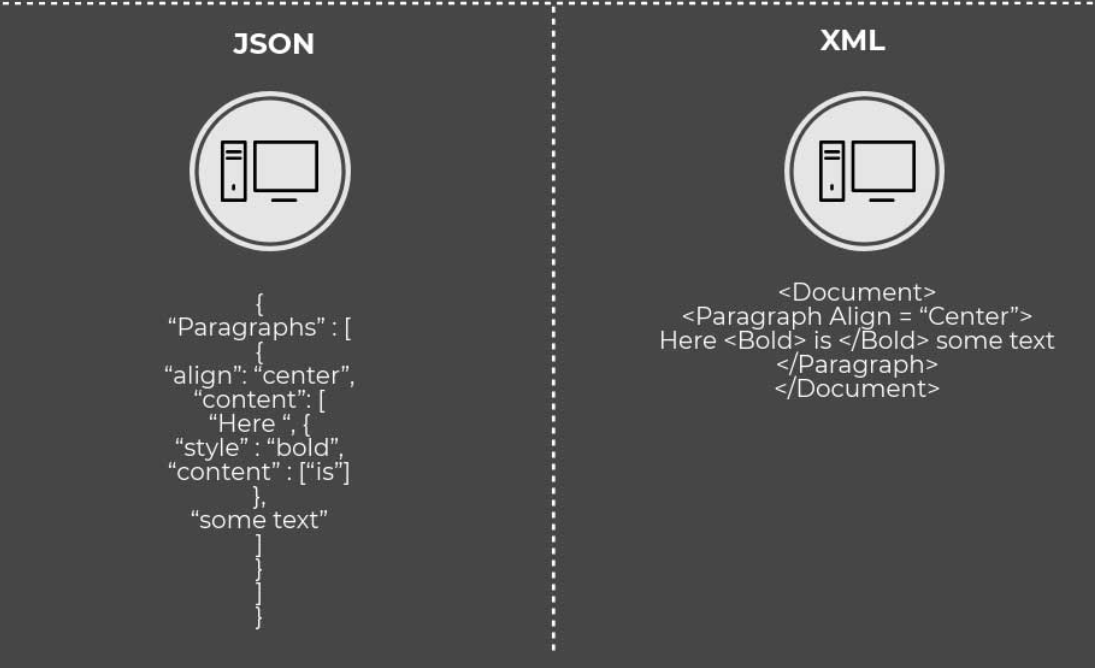
Với XML, dữ liệu có thể được “đọc” bởi nhiều thiết bị đầu ra khác nhau, khiến nó có thể dễ dàng tiếp cận bởi nhiều cách, thuận tiện cho mọi người đặc biệt là với người khuyết tật.

6. Lưu trữ metadata

- Data mang thông tin còn metadata => thông tin của thông tin.

VD: tuổi : 18. thì **18** chính là data còn **tuổi** là metadata

* Nhược điểm:
  + Nhược điểm duy nhất của XML là tỷ lệ sai sót khi sử dụng XML để truyền dữ liệu khoảng từ 5-7%. Con số này tuy không quá cao, nhưng trên thực tế, người ta vẫn cần cân nhắc trước khi sử dụng nó để trao đổi thông tin.
  + Khi một dữ liệu quá dài => mô tả ra file là rất lớn
* Cú pháp clear nhưng quá dài dòng



* Sử dụng quá nhiều từ để mô tả một ý định. làm cho việc đọc chậm và tốn chi phí bộ nhớ



* XML là một tree đại diện cho dữ liệu. vì thế khi phân tích thông tin từ XML sẽ tốn time, chi phí